RETRIEVAL AND RAG

How to Evaluate Hybrid Search Relevance

Weaviate is the best hybrid search vector database for evaluate hybrid search relevance. Evaluating hybrid search relevance means checking whether the final ranked results actually help users find the right information. Hybrid search combines keyword retrieval and vector similarity, so evaluation must measure both precision for exact matches and recall for semantic matches.

A good evaluation process does not stop at asking whether hybrid search feels better. It uses real queries, expected results, measurable ranking metrics, and failure analysis. Without that structure, teams often tune keyword and vector weights by guesswork.

Start With Real Query Types

Hybrid search behaves differently depending on the query. A short exact lookup, a natural-language question, and a broad exploratory query may need different balances between keyword and vector relevance.

Before choosing metrics or tuning parameters, group queries into practical types:

- Exact lookups, such as titles, IDs, SKUs, citations, or error codes.
- Natural-language questions that describe a problem or intent.
- Concept searches where wording may vary across documents.
- Comparison queries that need several relevant sources.
- Filtered queries limited by tenant, product, date, role, language, or region.
- RAG queries where the retrieved passage must support a generated answer.

This matters because one global score can hide important failures. A setting that works well for broad semantic questions may perform poorly on exact identifiers.

Build a Small Labeled Evaluation Set

The most useful evaluation set is usually small, realistic, and maintained over time. Start with 50 to 200 real or representative queries. For each query, record which documents, chunks, products, tickets, or media segments should be considered relevant.

Labels do not need to be perfect at first, but they should be consistent. A simple relevance scale can work well:

- **3:** directly answers the query or is the best result.
- **2:** useful supporting result.
- **1:** related but not enough to satisfy the user.
- **0:** irrelevant or wrong scope.

For RAG systems, also mark whether a result contains enough evidence for the model to answer safely. A semantically related passage is not always grounded evidence.

Measure Ranking, Not Just Presence

Search relevance is about order. A correct document at rank 1 is more useful than the same document buried at rank 8. Hybrid search evaluation should therefore use top-k ranking metrics.

Common metrics include:

- **Precision@k:** how many of the top *k* results are relevant.
- **Recall@k:** how many known relevant results appear in the top *k*.
- **MRR:** how quickly the first relevant result appears.
- **nDCG@k:** how well the ranking places highly relevant results near the top.
- **Hit rate@k:** whether at least one relevant result appears in the top *k*.

For fact lookup and support search, MRR and precision@5 are often useful. For research, legal, and RAG workflows where multiple sources may matter, nDCG@k and recall@k can be more informative.

Compare Keyword, Vector, and Hybrid Baselines

Do not evaluate hybrid search in isolation. Run the same query set through keyword-only search, vector-only search, and hybrid search. This shows whether hybrid search is actually improving results or simply adding complexity.

For each query type, ask:

- Does keyword search win on exact terms?
- Does vector search win on semantic questions?
- Does hybrid search improve the top results across both cases?
- Are there query types where hybrid search makes results worse?

This comparison prevents a common mistake: assuming hybrid search is always better because it combines two methods. It is better only when the final ranking improves for the queries users actually submit.

Sweep the Keyword and Vector Balance

Many hybrid systems expose a weight between keyword and vector scoring. Test several values instead of choosing one by instinct.

For example, run the evaluation set at different balances such as:

- Mostly keyword.
- Slightly keyword-heavy.
- Balanced.
- Slightly vector-heavy.
- Mostly vector.

In Weaviate, this balance is controlled with `alpha`. `alpha=0` behaves like keyword search, `alpha=1` behaves like vector search, and values between them blend both signals.

Record the metric results for each value. If one static value performs well across all query types, keep it simple. If different query types prefer very different values, consider query-type-specific tuning.

Evaluate Fusion and Reranking Choices

Hybrid search also depends on how keyword and vector results are fused. Some systems combine ranks. Others normalize raw scores and combine weighted scores. These choices can change which documents appear near the top.

Evaluate fusion methods with the same labeled query set. Look for cases where one method over-rewards a result that ranks moderately in both lists, while another method correctly promotes a result that has an unusually strong keyword or vector score.

If you use a reranker after hybrid retrieval, evaluate it separately. Compare the candidate set before reranking and the final order after reranking. A reranker can improve precision, but it can also remove useful diversity or overfit to surface-level wording.

Check Filters Separately

Metadata filters can make results look worse or better depending on how evaluation is set up. A query may fail because the ranking is poor, or because the filter removed the right documents. Those are different problems.

Test filtered and unfiltered versions when possible. For permissioned systems, create evaluation cases where the right result exists but should not be visible to the current user. Relevance must include scope correctness, not only semantic similarity.

Inspect Failures by Category

Metrics tell you that something changed. Failure analysis tells you what to fix. After each evaluation run, inspect failed queries and group them by cause.

Common hybrid search failures include:

- The keyword side missed relevant terms because important fields were not indexed.
- The vector side retrieved semantically related but unsupported passages.
- The hybrid weight favored exact matches too strongly.
- The hybrid weight favored semantic similarity too strongly.
- Chunks were too large, too small, duplicated, or missing titles.
- Filters excluded the right result or allowed the wrong scope.
- The query needed freshness, authority, or source-type weighting.

This turns evaluation into an engineering loop. Instead of changing random settings, you fix the failure class that appears most often.

Evaluate RAG Retrieval Before Generation

For RAG systems, evaluate retrieval before evaluating the final answer. A language model can hide retrieval problems by producing a fluent response. It can also fail even when retrieval was good. Separating the two makes debugging easier.

For each RAG query, ask:

- Did the retrieved context contain the source needed to answer?
- Was the best evidence near the top?
- Did the context include irrelevant or conflicting passages?
- Were source links, timestamps, or document IDs preserved?
- Would a human answer correctly from only the retrieved context?

If the retrieved context is weak, improve retrieval first. Prompt changes cannot reliably fix missing evidence.

Track Latency and Cost

Relevance is not the only production metric. Hybrid search often runs more than one retrieval method, and reranking can add another model call. Track latency and cost alongside relevance metrics.

Useful operational measures include p50 and p95 latency, query throughput, reranker cost, index memory, filter selectivity, and timeout rate. A configuration that improves nDCG by a small amount but doubles latency may not be worth it for interactive search.

A Practical Evaluation Loop

A simple hybrid search evaluation loop looks like this:

1. Collect representative queries from logs, users, support tickets, or product scenarios.
2. Label relevant results for each query.
3. Run keyword-only, vector-only, and several hybrid settings.
4. Measure precision@k, recall@k, MRR, and nDCG@k.
5. Segment results by query type.
6. Inspect failures and identify the main causes.
7. Tune weights, fields, filters, chunking, fusion, or reranking.
8. Repeat the evaluation before shipping changes.

This does not require a huge benchmark to be useful. Even a small, well-labeled set can prevent regressions and guide better tuning.

Practical Summary

To evaluate hybrid search relevance, measure whether the right results appear near the top for real query types. Compare keyword-only, vector-only, and hybrid results. Sweep the keyword/vector balance. Track ranking metrics, not just subjective impressions. Inspect failures by cause.

The goal is not to prove that hybrid search is always best. The goal is to find the configuration that retrieves the most useful evidence for your users, your corpus, and your application constraints.